Středoškolská technika 2022

Setkání a prezentace prací středoškolských studentů na ČVUT

Robot ovládaný mikrokontrolérem ESP32

Štěpán Kraus

Gymnázium Václava Hlavatého

Poděbradova 661, Louny

Prohlášení

Prohlašuji, že jsem svou práci SOČ vypracoval/a samostatně a použil/a jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Lounech dne 3. 4. 2023 

Štěpán Kraus

Poděkování

Děkuji panu Mgr. Jiřímu Studničkovi a Mgr. Drahomíru Zrůstovi za mé znalosti v tomto oboru.
A děkuji své rodině za podporu mého vzdělání.

Anotace

Tato práce se zabývá využitím mikrokontrolérů ESP32 v robotice. Popisuje části, zapojení a princip ESP32. Zaměřuje se na funkce a možnosti toho mikrokontroléru. Dále zjednodušeně popisuje význam a využití robotiky v praxi a funkčnost servomotorů. V druhé části se zabývá mou prací s praktickým využitím ESP32 v robotice.

Klíčová slova

ESP32; mikrokontrolér; robot; robotika

Annotation

This work looks at the use of ESP32 microcontrollers in robotics. It describes the parts, pinout and principle of the ESP32. It focuses on the functions and capabilities of that microcontroller. It further describes in simplistic terms the importance and use of robotics in practice and the functionality of servo motors. In the second part, it discusses my work with practical use of ESP32 in robotics.

Keywords

ESP32; microcontroller; robot; robotics

Obsah

ÚVOD

Robotika je věda ležící na průsečíku mnoha vědních předmětů - počínaje matematikou přes fyziku, výpočetní techniku, logiku až třeba k biologií (robotické ruce), je důležitou součástí pokroku a setkáváme se s ní čím dál tím více. I když se s robotikou nevídáme každý den, je to důležitá součást průmyslu a výroby. Se zdokonalováním automatizace a příchodem Průmyslu 4.0 se tato věda stává rozvinutější a otevřenější.

Roboty ovšem mohou být ledajací, od těžkých průmyslových svářecích robotů přes roboty manipulující se skalpely na operačním sále až po roboty, kteří vám uvaří kávu a vyčistí kávovar. Se zdokonalováním umělé inteligence se robotika posouvá na další úroveň, kdy stroje nejen že budou moci vykonávat pro člověka nebezpečné činnosti, ale budou moci i sami se rozhodovat v některých situacích, kde nebude jen jedna cesta, jak daný úkol splnit.

Všechny tyto roboty ovšem musí něco ovládat, a tato práce se zabývá převážně mikrokontrolérem ESP32 jakožto „mozkem“ robota, a to jak z hlediska hardwaru, tak jeho využitím a funkcemi.

ESP32 je takový malý pomocník, který dokáže velké věci, převážně všestrannost tohoto mikrokontroléru umožňuje realizaci mnoha nápadů a vytváření věcí s praktickým využitím.

Práce také rozebírá pohyb robotů, podrobně popisuje servomotory a jejich praktické využití. Dále také komunikaci prostřednictvím IoT (Internetu věcí) a pulzně šířkové modulace. V neposlední řadě tato práce popisuje některé součásti, užití a činnosti průmyslových robotů.

Cílem praktické části práce je pokus o navrhnutí, vymodelování, vytisknutí, naprogramování a rozpohybování čtyřnohého robota řízeného mikrokontrolérem ESP32 s užitím bezplatných programů.

Jedná se o kvadruped poháněný 12 servomotory, programovaný v jazyce C++, v prostředí Arduino IDE, modelovaný v programu Blender, osazený ultrazvukovým senzorem pro detekci překážek a displejem pro vyjádření „emocí“ robota.

1 TEORETICKÁ ČÁST

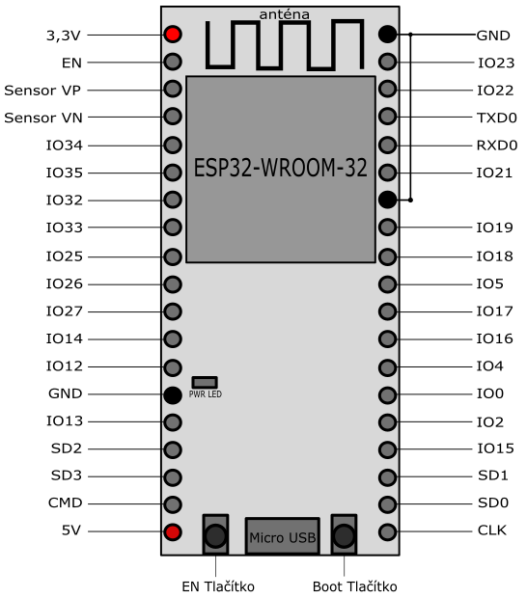
1.1 ESP32

ESP32 je jeden z mnoha mikrokontrolérů využívaných v této době nejen v praxi, ale i na výuku robotiky. Má své klady i zápory, Největší rozdíl od jiných mikrokontrolérů, který se dá lehce zpozorovat je cena, která je vzhledem k možnostem tohoto zařízení velmi nízká, na rozdíl od konkurence. Vyniká především podporou komunikace přes Wi-Fi a Bluetooth 4.0 LE, tudíž je platformou pro aplikace na bázi internetů věcí (Internet of Things – dále „IoT“). V této práci se zabývám převážně modulem ESP-WROOM-32

1.1.1 ESP-WROOM-32 – Hardware a piny

- 34 x GPIO piny, 16 x podporující PWM
- 18 x 12-bit A/D převodníky
- 2 x 8-bit D/A převodníky
- 10 x dotykové sensory
- 3 x SPI – sériové rozhraní a 2 x I2S – sériové rozhraní
- 2 x I²C – sériové rozhraní s maximální frekvencí 5MHz
- 3x UART – sériové rozhraní

[1]



1: Stručný Popis ESP32

1.1.2 Další hardware mikrokontroléru

Mikrokontrolér je osazen dvoujádrovým 32-bitovým procesorem, pamětí SRAM o velikosti 520 KB, oscilátory pro připojení přes Wi-Fi a Bluetooth a časovačem [2].

1.1.3 GPIO piny

Na mikrokontroléru ESP32 se nachází 34 GPIO pinů. Jedná se o univerzální vstupní/výstupní piny, přes které se dá ovládat nespočetně mnoho věcí, ale hlavně se dají použít i jako vstup pro senzory a vypínače, pomocí kterých se dokáže ESP32 dále řídit.

1.2 Mikrokontrolér

Mikro – malý, kontrolér – ovladač, tato definice je zcela výstižná, mikrokontrolér je malé zařízení, které dokáže ovládat malé i velké věci. Mezi ty nejznámější patří Arduino a ESP32. Do češtiny se často také překládá jako Jednočipový počítač. Tyto Jednočipové počítače existují v mnoha variantách. K nim jsou také vytvořené moduly, kterými se dají rozšiřovat funkce těchto počítačů.

1.3 Robotika

Věda o robotech neboli robotika se zabývá designem, funkčností a autonomií již zmiňovaných robotů. Úzce spolupracuje s elektronikou, mechanikou a vědami zabývajícími se softwarem. Nejpopulárnější oblastí této vědy je průmyslová robotika, do které spadají průmyslové stroje s několika pažemi, autonomií a univerzálností. Robotika má veliký potenciál do budoucnosti v oblasti automatické výroby, průmyslu 4.0 a například i v řešení nebezpečných situací, u kterých hrozí ohrožení lidských životů, jako je zneškodňování bomb, práce s radioaktivními prvky a materiály atd...

1.3.1 Robot

Robot je stroj či systém, který je umělého původu, je schopen manipulovat s věcmi, je opakovaně programovatelný a je schopen pohybu [3]. Slovo robot je českého původu a proslavilo se po celém světě. V dnešní době se můžeme setkat s nekonečně mnoha roboty,

které vykonávají cokoli, na co jen pomyslíme. Existují také takzvané kolaborativní roboty, které dokáží pracovat precizně a bezpečně, a to hned vedle člověka.

Roboty mají také rozdělení podle počtu podpěrných končetin (pokud nějaké mají).

- Bipedální – dvě končetiny
- Tripedální – tři končetiny
- Kvadrupedální – čtyři končetiny
- A tak dále…

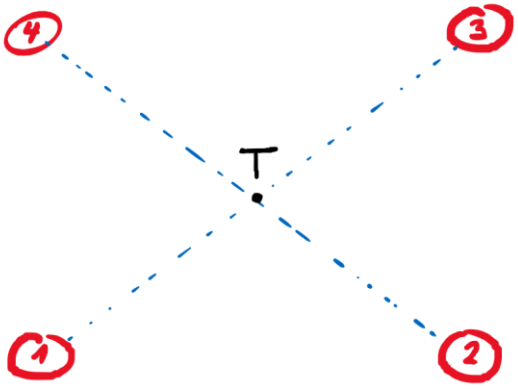
Nejčastěji se ovšem setkáme s roboty bipedálními, kvadrupedálními a hexapedálními.

1.3.2 Teorie chůze robota

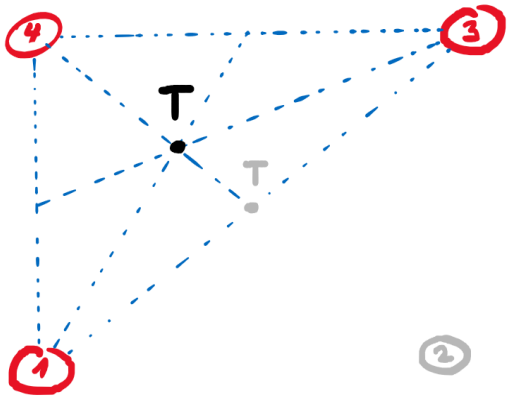
Teorie chůze robota je prostá - těžiště robota se musí nacházet ve středu obrazce vytvořeného propojením bodů dotýkajících se země. Neboli - když budeme mít kvadrupedálního robota a obrazcem bodů dotýkajících se země bude čtverec, těžiště tohoto robota musí být kdekoliv nad průsečíkem úhlopříček obrazce. Tím by náš robot neměl ztratit stabilitu. (obrázek 2)

Kdybychom výše uvedeného robota chtěli uvést do pohybu tím, že zvedneme jednu z nohou, je potřeba přenést těžiště do nově zobrazeného obrazce. (obrázek 3)

Tímto by se náš robot měl postavit pouze na 3 ze 4 nohou a zůstat stále stabilní.



2: Nákres teorie chůze robota 1



3: Nákres teorie chůze robota 2

1.3.3 Průmyslové roboty

Roboty používané v průmyslu většinou nemají končetiny na pohyb, občas se setkáme s koly, nýbrž vynikají pažemi. Většinou co robot, to paže, ale můžeme se setkat i s roboty, které mají více paží.

Průmyslové roboty se rozdělují na tři kategorie:

- Svařovací
- Lakovací
- Paletizační

Svařovací roboty, jak již vyplývá z názvu, sváří kovy, a to s dokonalou precizností, a zvládají téměř všechny druhy svařování. Výhodou těchto robotů je převážně jejich bezpečnost.

Lakovací roboty mají využití převážně v automobilovém průmyslu k lakování aut, ovšem dají se využít i v průmyslu stavebním k lakování nábytku apod.

Paletizační roboty se starají o skládání a přesun palet, tyto stroje jsou schopny unést i těleso o hmotnosti dvě tuny. Nejčastěji se s těmito roboty setkáme v potravinářském průmyslu. [4]

1.4 ESP32 v robotice

Mikrokontrolér ESP32 díky svým 34 pinům podporujícím PWM neboli pulzně šířkovou modulaci, dokáže ovládat servomotory. V robotice narazíme na servomotory poměrně často z důvodu, že servomotory díky své přesnosti a rychlosti, která je v mnoha případech lepší než u lidí, zvládají otáčení do přesně zadaných pozic, a to v relativně krátkém čase (podrobně popsané v části 1.4).

Ovšem servomotory nejsou to jediné o co v robotice jde a co ESP32 dokáže ovládat, dále to jsou také displeje, bzučáky, různé snímače a mnoho dalších. Nicméně nejdůležitější funkcí zůstává přenos dat přes síť Wi-Fi a přes Bluetooth. Díky tomuto přenosu se dá sledovat stav robota na dálku, případně různé hodnoty ze snímačů, a dokonce se dá i bezkontaktně ovládat například přes mobilní aplikaci či webové prostředí vygenerované samotným mikrokontrolérem.

ESP32 je takový všestranný kompaktní pomocník, který dokáže řídit velké věci. Existují ale například i modely s tímto modulem a kamerou, kde při správném naprogramování se dá klidně z tohoto 5 centimetrového „cvrčka“ udělat webkamera, která sdílí obraz v reálném čase a k tomu může i například rozpoznávat obličeje pro další použití.

1.4.1 Webový server s ESP32

Webový server je kombinace hardwaru a softwaru, která se stará o správu a předávání webových stránek klientům. ESP32 se díky podpoře IoT může stát takovým serverem. Informace předané ze serveru (ESP32) klientovi mohou mít více podob, a to například: obrázky, text, videa, či dokument psaný v HTML.

HTTP neboli Hyper Text Transfer Protocol - je protokol odpovědný za komunikaci serveru s klientem. V tomto typu komunikace si klient vyžádá data a server mu je následně přes stejný protokol poskytne.

Komunikace serveru a klienta je prostá: klient (v našem případě webové rozhraní chytrého telefonu) požádá o načtení webové stránky jakožto dokumentu v HTML. ESP32 poté zašle a zobrazí požadovaný dokument. Klient pak při jakékoliv operaci v dokumentu (stisknutí tlačítka atd…) odesílá požadavek o provedení akce spojené s daným tlačítkem. ESP32 požadavek přijme a akci vykoná.

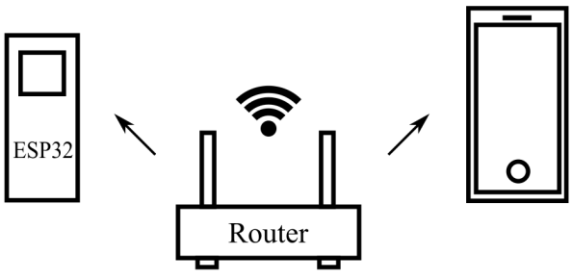
ESP32 může komunikovat ve 3 různých režimech, a to je:

- Station Mode (STA)
- Soft Access Point Mode (AP)
- STA + AP Mode

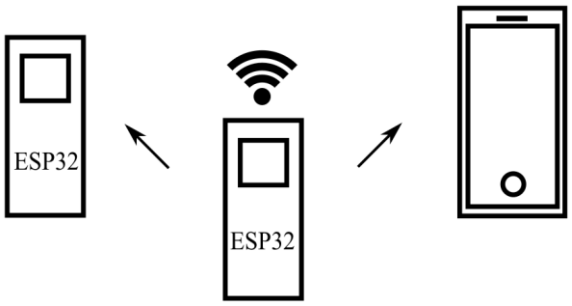
Station Mode neboli režim stanice je režim, ve kterém se ESP32 připojí na již existující síť Wi-Fi, stejně jako například naše mobilní telefony a notebooky. ESP32 se připojí přes SSID routeru a přes heslo, následně mu je připsána IP adresa.

Soft Access Point Mode je režim, ve kterém si ESP32 vytvoří svojí vlastní síť Wi-Fi, stejně jako router, čímž umožní ostatním zařízením se připojit na danou síť. Z důvodu toho, že ESP32 nemá port na Ethernetový kabel, tento režim se nazývá Soft AP. Je potřeba v programu nastavit SSID a heslo, přes které se zařízení jako telefony, notebooky apod. budou připojovat.

Režim STA + AP Mode je kombinace obojí, kde ESP32 funguje jako přístupový bod a zároveň se může připojit na jiná zařízení. [5]



4: ESP32 nákres režimu připojení STA



5: ESP32 nákres režimu připojení AP

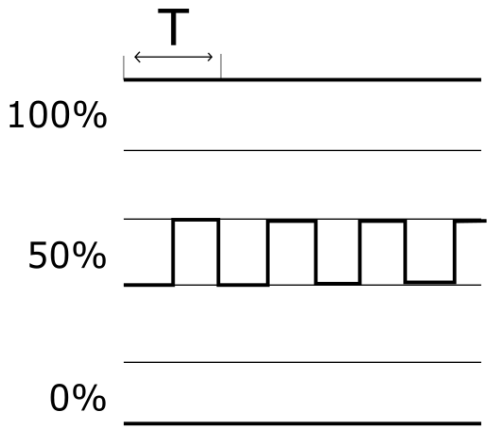
Konfigurace režimů je díky knihovně Wifi velmi jednoduchá, stačí ve funkci Start() použít funkci Wifi.mode(WIFI_x), za x pak dosadíme AP (přístupový bod), nebo STA (režim stanice).

Díky těmto funkcím můžeme vytvořit webové prostředí pro ovládání robotů odkudkoliv, či vytvořit uživatelsky přívětivé prostředí, pomocí kterého bude moct robota jednoduše ovládat.

1.4.2 Pulzně šířková modulace

Pulzně šířková modulace (PWM – z angličtiny Puls Width Modulation) je způsob přenosu dat z vysílače na přijímač, pomocí krátkých střídavých změn signálu. Signál zde má konstantní periodu T, ve které se mění stav signálu v závislosti na určeném poměru.

V našem případě vysílačem je ESP32 a přijímačem jsou servomotory na nohou robota. Uvnitř servomotorů se nachází elektronika, která dokáže PWM signál rozklíčovat a v závislosti na hodnotě (%) pak otočit servomotor do dané polohy.



6: Demonstrace PWM signálu

Pomocí pulzně šířkové modulace se ovšem nedají ovládat jen servomotory, ale také například svítivé diody, kde pomocí PWM nastavujeme svítivost diod, což se dá využít například v dálkově ovládaných zařízeních za pomoci IR přenosu.

1.5 Servomotory

Servomotor má od běžných motorů velikou výhodu, dá se nastavovat poloha jeho osy. U servomotorů se setkáme především se dvěma typy: 180 ° a 360 °. Jejich rozdělení odpovídá za vše, u jedněch je osa omezena pouze na pohyb 0 ° až 180 °, u těch ostatních není osa omezena vůbec, tudíž se dá otáčet o celých 360 °. V robotice se nejčastěji setkáme s elektrickými servomotory, které se ovládají pomocí výše zmíněného PWM signálu, který udává o kolik stupňů se má osa servomotoru otočit.

Servomotory se využívají v mnoha zařízeních kolem nás, převážně ale v robotech pracujících s přesností.

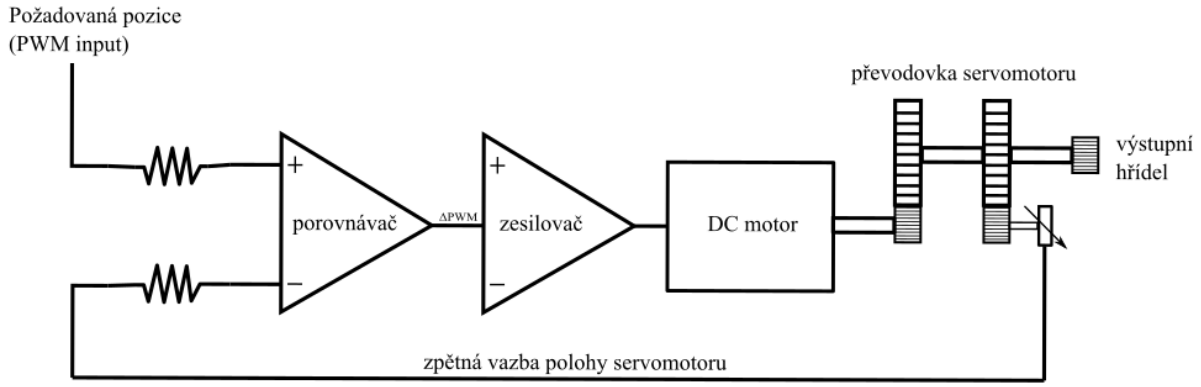
1.5.1 Servomotory SG90 a MG995

Servomotory SG90 a MG995 jsou nejčastěji používány v lehké robotice a v modelech na dálkové ovládání. SG90 je celoplastový servomotor a výhodou je, že moc neváží a zároveň dokáže vyvinout velkou sílu. MG995 je o něco větší plastový servomotor, ovšem s železnou převodovkou, tudíž se jen tak nezničí a má daleko větší sílu. Výhodou obou modelů je dále cena, která se pohybuje v rozmezí 60 až 300 korun českých, cena lepších motorů se dokáže vyšplhat až na řády tisíců, či statisíců.

1.5.2 Ovládání servomotorů

Elektrické servomotory se ve většině případů ovládají pomocí signálu PWM.

Potenciometr napojený osou či převodovkou na motor, snímá polohu motoru a odesílá zpětnou vazbu (signál PWM), který se poté porovnává se signálem dodaným z mikrokontroléru a vzniklá odchylka (rozdíl signálů PWM) je převedena na vyšší napětí, jež je následně dodáno motoru, který se otočí do mikrokontrolérem zadané polohy. Existují i servomotory s jiným způsobem snímání - ovšem v lehké robotice, kterou se tato práce převážně zabývá, se používá nejčastěji tento způsob.



7: Stručný diagram elektroniky servomotoru

Nevýhodou levných servomotorů je, že při výrobě často vzniká odchylka, ovšem ta se dá poté eliminovat v programu robota.

Servomotory se ve většině případů vyrábí s ozubeným kolečkem na hřídeli a dodává se s nástavci, které se za pomoci šroubu přichytí na hřídel servomotoru. Tyto nástavce se následně dají připevnit k mechanismu a servomotor jím může snadno otáčet.

1.6 Ovládání robotů

Ovládání robotů ať už s ESP32 či s jiným mikrokontrolérem je velmi rozmanité. Není těžké vytvořit aplikaci, která je uživatelsky přívětivá, a se kterou se snadno ovládá i ten nejsložitější robot. Nevýhoda pouze je, že taková aplikace zabere na vytvoření nemálo času.

Řízení robotů může být například prostřednictvím webové či mobilní aplikace, ale také vznikají programy a celé sítě softwarů pro ovládání několika desítek robotů zároveň. Jednou z variant ovládání robotů může být tzv. „Digitální dvojče“ neboli program, ve kterém je robot vyobrazen jakožto 3D model, vy jen určíte polohu hlavice a zbytek udělá software za vás. Tento software se pohybuje v řádech stotisíců korun českých za licenci, a to z důvodu, že tyto roboty se využívají ve firmách a na výrobních linkách, kde jejich přesnost a snadné naprogramování je nejdůležitější část. [6]

U robotů jako je ten, kterým se zabývám v praktické části se používají většinou aplikace, buď vytvořené přímo vývojářem robota, nebo univerzální aplikace pro ovládání robotů.

Samozřejmostí je také mechanické ovládání robotů prostřednictvím ovladačů, či ovládacích panelů s vypínači, tlačítky apod. Nejčastěji se ovšem v robotice setkáme s ovládacími panely připomínající tablet, který má v sobě nahranou aplikaci nebo nějaký software, kterým se dá

robot lehce ovládat. Tyto aplikace jsou buď vytvořeny do prostředí nějakého z operačních systému (Windows, Android, IOS), nebo jsou přímo samostatné, anebo jsou vytvořené na bázi webového serveru, aby byly přístupné kdykoliv odkudkoliv.

V dnešní době se vývojáři ovladačů a programů ovládajících roboty snaží vytvořit co nejvíce uživatelsky přívětivé, jednoduché a intuitivní ovládání, aby se dokázali lidé spravující tyto roboty lehce naučit jejich ovládání.

Existují i roboty ovládané samostatně, bez pomoci člověka, nebo ovládané umělou inteligencí, ty pak jen stačí zapnout a dělají, co dělat mají.

1.7 Internet věcí

Internet věcí (neboli IoT) je nový trend v oblasti komunikace, týkající se internetu a spolupráce člověka s technologií a propojení různých zařízení. Tímto se zabývá průmysl 4.0, kde je snaha o autonomní továrny, které by dokázaly pracovat zcela samostatně za pomoci IoT. Jedná se například o chytrý koš, který si při naplnění sám zavolá popeláře a sám se vynese. Pro mnohé lidi může být tato vidina odstrašující, ale z hlediska úrazů na pracovišti a bezpečnosti lidí v těžkém průmyslu to zas tak špatné není.

V dnešní době se IoT a průmysl 4.0 zaměřuje na umělou inteligenci, cloudové uložiště a zpracovávání velikého množství dat. Jedná se o technologii, která jde neustále dopředu a vývoj této technologie se stal jednou z priorit vědců. [7]

ESP32 má podporu IoT a dokáže vytvořit webové prostředí a propojit se prostřednictvím API (Application Programming Interface), skrz které dokáže sbírat a zpracovávat data z jiných webových stránek a aplikací na internetu. [8]

ESP32 existuje dokonce i s modulem CAM, což je model s několika GPIO piny a kamerou, pomocí které dokáže ESP32 vysílat živě záznam z kamery ve webovém prostředí odkud se dá sledovat na jakémkoliv zařízení s přístupem na internet. Dále tento modul má zabudovanou podporu umělé inteligence, která může například rozpoznávat obličeje a podle výsledku se dále rozhodovat.

2 PRAKTICKÁ ČÁST

Praktická část mé práce se zabývá mnou sestaveným čtyřnohým robotem s mikrokontrolérem ESP32. Jedná se tedy o malého robota, který se skládá z mnou softwarově vymodelovaných a následně vytisknutých na 3D tiskárně dílů a elektronických součástek, které se dají běžně sehnat.

2.1 Základní popis robota

Jedná se o dálkově ovládaný quadruped neboli čtyřnohého robota, jenž je osazen mikrokontrolérem ESP32 jakožto řídící jednotkou a zařízením, díky kterému se robot dá ovládat na dálku. Na každé ze čtyř nohou se nachází tři servomotory. Tyto servomotory jsou napojeny na modul PCA9685, který podle vstupu z mikrokontroléru adresně posílá PWM signál až 16 servomotorům.

2.2 Hlavní body práce

Nejdříve bylo potřebné robota vymyslet. K tomu postačila obyčejná tužka a papír a po několika návrzích vznikl první nákres. Při vytváření nákresu jsem se inspiroval z internetu. [9]

Dalším krokem byla modelace dílů a schéma zapojení elektroniky. Pro modelaci a úpravu plastových částí byl použit program s názvem Blender, jedná se o tzv. „freeware“ (bezplatný software) volně ke stažení na internetu. Dále pro schéma zapojení bylo použito webové rozhraní Tinkercad a program Fritzing. Ve všech zmiňovaných programech jsem se učil tvořit převážně za pomoci návodů na platformě YouTube.

2.2.1 Modelace částí

Pro modelaci byl použit již zmíněný Blender. Zde za pomoci stereometrie a 3D prostoru se podařilo později vymodelovat celého robota v měřítku 1:1. Nejdříve se začalo servomotory, protože mají dané rozměry, které se použily a vymodelovaly přesně podle předlohy. Poté již stačilo kolem motorů vytvořit konstrukci nohou a těla, které drželo nohy pohromadě.

Modelace částí zabrala nejvíce času dohromady s tiskem, převážně z důvodu neustálých úprav a vylepšování součástí tak, aby dané součásti vydržely pod tíhou těla robota a aby se nezlámaly při pohybu.

2.2.2 Tisk součástí robota

Celkem se každá část robota tiskla minimálně 2x. Tisk částí robota celkově zabral něco přes 20 hodin. Po vytisknutí všech částí ovšem bylo třeba za pomoci 3D pera zpevnit některé části (převážně v kloubech). Osvědčila se i metoda filamentu a zapalovače, ale z důvodu bezpečnosti bylo častěji použito 3D pero. Nejvíce však celému modelování a tisknutí pomohla modelace robota v měřítku 1:1, neboť při vylepšování částí se nemusel brát ohled na velikost, jen čistě na tvar.

Největší výhodou 3D tisknutých částí je jejich váha, pohybují se v řádech gramů, za to snesou poměrně velkou zátěž, a to až do teploty 60°, při kterých se vytisknuté části začínají ohýbat.

2.2.3 Schéma zapojení elektroniky

Pro vytvoření schématu sice mohla být použita jen tužku a papír, ale práce v softwarech jako je Fritzing a Tinkercad je daleko pohodlnější. Ovšem mým prvním nástrojem se stal Microsoft Whiteboard, což je aplikace na počítač, která funguje stejně jako mazací tabule například ve škole.

Ze znalostí z hodin fyziky a za pomoci internetu jsem poté byl schopen dát dohromady první nákres obvodu pro robota.

2.3 PCA9685

Modul PCA9685 je shield na ovládání servomotorů až na 16 kanálech. Nejčastěji se používá s mikrokontroléry, které nemají dostatečný počet GPIO pinů s podporou PWM. V mém případě slouží pouze jako usnadnění práce. Další výhodou tohoto modulu je tzv. „cable management“, všech 16 pinů ovládajících servomotory je seřazeno vedle sebe, čímž nevzniká ve vodičích chaos a lépe se v nich orientuje.

Modul se cenově pohybuje mezi těmi levnějšími a k sehnání je ve většině prodejen a internetových obchodů s elektronikou.

Ke komunikaci mikrokontroléru s modulem je použita metoda I^2C. Modul je připojen přes 2 piny, a to SCL a SDA. Dále se zde nachází pin kladného pólu VCC a přídavného napětí V+ a samozřejmostí je pin na záporného pólu GND.

2.4 Programování robota

Pro programování robota jsem zvolil prostředí Arduino IDE neboli prostředí pro programování čehokoliv s mikrokontrolérem z řad Arduino nebo ESP. Při programování ESP-32 v tomto prostředí je použit jazyk C++.

Programovací jazyk C++ patří mezi nejrozšířenější jazyky vůbec, hlavně díky podpoře několika stylů programování. Jméno tohoto jazyka je odvozeno od jeho předchůdce C společně s operátorem inkrementace v tomto jazyce neboli „++“. Z hlediska náročnosti C++ patří mezi středně náročné jazyky na pochopení. Velice se podobá, a dokonce i dokáže společně fungovat se svým předchůdcem C, kde mnoho věcí a pravidel (převážně v kontextu) zůstalo.

Arduino IDE je prostředí velmi příjemné pro jak začátečníky, tak pokročilé, má mnoho užitečných funkcí v uživatelsky přívětivém provedení a neskutečně mnoho návodů na ovládání. Všechny níže uvedené výstřižky z programu jsou vytvořeny v aplikaci Arduino IDE.

2.4.1 Popis základního programu

K základnímu naprogramování robota byly využity dvě knihovny, a to Wire a Adafruit_PWMServoDriver. Knihovna Wire byla použita ke komunikaci prostřednictvím I^2C pinů a knihovna Adafruit_PWMServoDriver byla využita k ovládání servomotorů pomocí modulu PCA9685.

Nejprve bylo potřeba definovat použité knihovny:

```
1  #include <Wire.h>
2
3  #include <Adafruit_PWMServoDriver.h>
4
```

8: Definice použitých knihoven

Poté bylo potřeba definovat na jaké adrese v I^2C komunikaci se nachází modul:

```
6  Adafruit_PWMServoDriver board1 = Adafruit_PWMServoDriver(0x40);
```

9: Definice adresy modulu PCA9685

Adresa modulu je vždy daná výrobcem modulu a je třeba jí vyhledat například v příručce. V tomto případě se jedná o adresu 0x40. Modul PCA9685 je v horní části programu definován s názvem „board1“.

V knihovně Adafruit_PWMServoDriver se nachází funkce s názvem setPWM(), která nastaví signál PWM na daném pinu. Nastavení signálu má tři parametry, a to pin, čas začátku a čas konce zaslaného pulzu. U servomotorů použitých na tomto robotovi je čas začátku 0, tudíž tento parametr můžeme vynechat a čas konce může být libovolné číslo od 0 do 4096, ovšem tyto servomotory fungují na intervalu 125 až 575 (0 ° - 180 °), a pro snadné programování je zde vytvořena funkce uhelnapulz(), která namapuje interval úhlů na interval pulzů:

```
82  int uhelnapulz(int uhel){
83      int pulz = map(uhel,0, 180, SERVOMIN,SERVOMAX);
84      return pulz;
85  }
```

10: Program pro mapování intervalů úhlu a velikosti pulzu

Proměnné SERVOMIN a SERVOMAX jsou definovány v horní části a nastaveny na hodnoty 125 a 575. Namísto těchto hodnot by se daly použít celá čísla, ovšem tímto způsobem program vypadá čistěji a lépe.

Důležité je opravdu do této funkce vkládat pouze hodnoty odpovídající intervalu <0 °; 180 °>, při dosazení jiných hodnot by program nemusel správně fungovat.

Ještě aby v tom byl úplný pořádek, na začátku programu jsou definovány názvy servomotorů a piny, na kterých dané servomotory jsou připojeny:

```
16  int LRhip = 0;
17  int LRknee = 1;
18  int LRankle = 2;
19
20  int RRhip = 4;
21  int RRknee = 5;
22  int RRankle = 6;
23
24  int LFhip = 10;
25  int LFknee = 9;
26  int LFankle = 8;
27
28  int RFhip = 12;
29  int RFknee = 13;
30  int RFankle = 14;
```

11: Definování pinů a názvů pro servomotory

Jedná se tedy o dvanáct servomotorů (tři na každé noze), nazvaných dle jejich polohy:

- Kyčel – hip
- Koleno – knee
- Kotník – ankle

Samozřejmě daly by se použít jakékoliv názvy, ale tyto jsou krátké, přehledné a dostačující.

Dále se u jmen servomotorů nachází zkratky nohy, na které se servomotor nachází neboli:

- LR – Left rear (levá zadní)
- RR – Right rear (pravá zadní)
- LF – Left front (levá přední)
- RF – Right front (pravá přední)

Tímto je snadné nalezení potřebného servomotoru, tudíž když bude potřeba nastavit kyčel na levé zadní noze robota do polohy 75°, program bude vypadat následovně:

board1.setPWM(LRhip, 75);

2.4.2 Popis programu StartPos()

Program StartPos() neboli program, který uvede robota do základní (startovní) polohy kdy robot stojí na všech čtyřech nohou a tělo má ve vzduchu je následující:

```
175  void StartPos()
176  {
177    board1.setPWM(LRhip, 0, uhelnapulz(70) );
178    board1.setPWM(LRknee, 0, uhelnapulz(120) );
179    board1.setPWM(LRankle, 0, uhelnapulz(180) );
180
181    board1.setPWM(RRhip, 0, uhelnapulz(100) );
182    board1.setPWM(RRknee, 0, uhelnapulz(120) );
183    board1.setPWM(RRankle, 0, uhelnapulz(60) );
184
185    board1.setPWM(LFhip, 0, uhelnapulz(90) );
186    board1.setPWM(LFknee, 0, uhelnapulz(100) );
187    board1.setPWM(LFankle, 0, uhelnapulz(140) );
188
189    board1.setPWM(RFhip, 0, uhelnapulz(90) );
190    board1.setPWM(RFknee, 0, uhelnapulz(160) );
191    board1.setPWM(RFankle, 0, uhelnapulz(75) );
192
193  }
```

12: Program StartPos()

Ještě před vytvořením tohoto programu bylo nutné zjistit polohy servomotorů v této základní pozici. Pokud by servomotory byly nastaveny na 90°, tato práce by zdaleka nezabrala tolik času. Jenže tím, že servomotory mají odchylku způsobenou při montáži robota, bylo hledání správného směru a pozice daleko náročnější. Poloha se dá najít metodou pokus omyl, kde stačí ten daný servomotor nastavit na 90°, čímž je ihned k vidění odchylka, a pak zbývá jen zjistit pomocí 100° a 80° kladný a záporný směr servomotoru. Vše pomocí příkazů setPWM() a uhelnapulz().

2.4.3 Pomalý pohyb servomotorů

Příkaz setPWM() je skvělý, až na jednu věc, nedá se nastavit rychlost jakou se servomotor do dané polohy otočí. Tady přichází na scénu funkce MoveSlow().

MoveSlow má dva parametry, a to pin servomotoru a úhel, neboli polohu, do které chceme servomotor dostat. Pro tuto funkci je ale také potřeba definovat aktuální polohu servomotoru. Je nekonečně mnoho způsobů, jak vytvořit tuto funkci, a s velkou pravděpodobností je mnoho snazších způsobů, ale zde byl zvolen tento:

1. Funkce si zjistí, o jaký servomotor se jedná a nastaví proměnnou currAngle (aktuální úhel) na aktuální úhel daného servomotoru.
2. Následně ve smyčce While(true){} zjistí, zda se servomotor bude pohybovat kladným nebo záporným směrem, a v závislosti na tom bude přidávat či ubírat velikost aktuálního úhlu a nastaví servomotor na pozici s novou hodnotou zvětšenou o velikost proměnné s názvem „speed“ (počet úhlů za 50 milisekund).
3. Mezi každým opakováním této smyčky je pauza 50 milisekund.
4. Pokud absolutní hodnota rozdílu aktuální velikosti úhlu a proměnné „speed“ je menší než „speed“, tak se program uvolní ze smyčky.

Tímto postupem dojdeme k tomu, že se dá regulovat rychlost servomotoru pomocí proměnné speed. Tato proměnná je definována na začátku programu v našem případě má hodnotu 2, tedy servomotor se otočí o 2° každých 50 milisekund. Jde i udělat varianta, kde ve spouštění příkazu se nastaví rychlost, ale v našem případě byla tato varianta zbytečná.

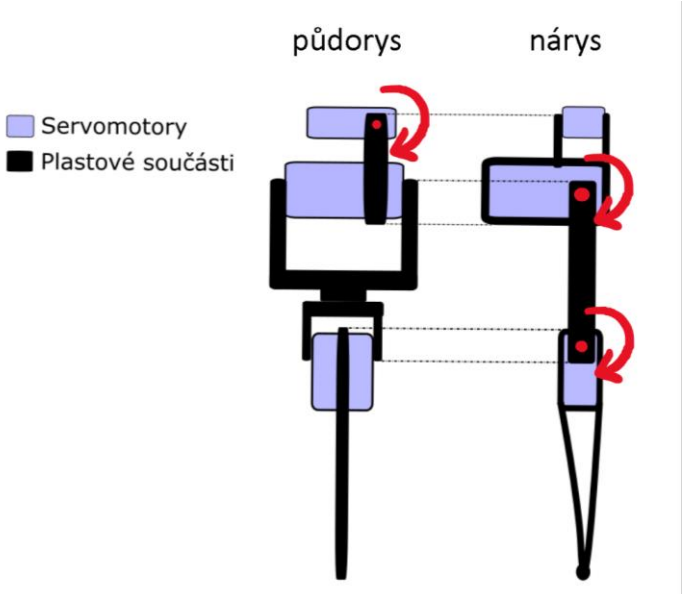
2.5 Chůze robota

Jak je již vysvětleno v teorii „chůze robota“ v teoretické části této práce, robot musí vždy být vyvážen. Pro šestinohého robota by to nebyl takový problém, tři nohy se zvednou, tři jsou podpůrné s těžištěm uprostřed. Ovšem u čtyřnohého robota (kvadrupedu) to je o něco složitější. Při zvednutí jedné ze čtyř nohou musí robot přenést těžiště na nohu protilehlou, jinak je zde velké riziko, že se robot převrátí.

Navíc u robota tohoto typu je také na obtíž tvar nohou. Byly zvoleny nohy se zakulaceným zakončením, což vede k tomu, že se robot při pohybu klouže. Tyto nohy se však dají kdykoliv nahradit nějakými s ostrým zakončením do špičky.

Vzor pohybu, neboli zvíře, které sloužilo k inspiraci pro vytvoření pohybu vpřed robota, posloužila želva. Přesněji dokument o želvách, které putují pouští, z čehož se dalo lehce zpozorovat, jak tento kvadrupedální pohyb funguje. Pokud vezmeme kartézskou soustavu souřadnic s vertikální osou Y, tak motory na nohou robota se pohybují tak, že kyčelní servomotory se otáčí kolem osy Y a kolenní a kotníkové servomotory se otáčí kolem osy X.

Zde je červeně znázorněn směr otáčení servomotorů:



13: Stručný popis nohou robota

2.6 Ultrazvukový senzor

Na robotovi se nachází ultrazvukový senzor HC-SR04, jedná se o jeden z cenově nejdostupnějších senzorů pro roboty. Funkce tohoto senzoru spočívá v měření vzdálenosti. Tento senzor funguje stejně jako echolokace u netopýrů či delfínů, tj. senzor vysílá zvukové vlny a podle doby, za kterou se zvuk vrátí zpět k senzoru, se určí vzdálenost. To se dá jednoduše vyjádřit vzorcem:

$$s = \frac{1}{2} t * c$$

V této rovnici: *s* je vzdálenost objektu od senzoru, *t* je čas, za který se zvuk vrátí (ovšem tam a zpět, tudíž pouze jedna polovina) a *c* je rychlost zvuku.

Za pomoci tohoto senzoru si robot může například ověřit, zda se před ním nenachází překážka, případně může reagovat na objekt vložený před něj. Senzor pracuje se zvukem na frekvenci vyšší jak 20kHz, tudíž pro lidské ucho je téměř nemožné, aby byl tento zvuk vnímán

Nevýhodou tohoto senzoru může být špatná orientace v prostředí s horší akustikou. Dále u levnějších typů tohoto modulu se můžeme setkat i s nepřesností, či anomáliemi při snímání.

2.6.1 *Programování senzoru HC-SR04*

Pro snadnější programování senzoru HC-SR04 pomocí ESP32, byla použita knihovna HCSR04.h [10]. Na začátku programu jsou definovány piny, na kterých jsou připojeny konektory „Trigger“ a „Echo“ (konektory zajištující odesílaní a přijímání zvuku senzorem).

```
8   UltraSonicDistanceSensor UZSenzor(0, 2);
```

14: Úryvek z kódu pro definici pinů ultrazvukového senzoru

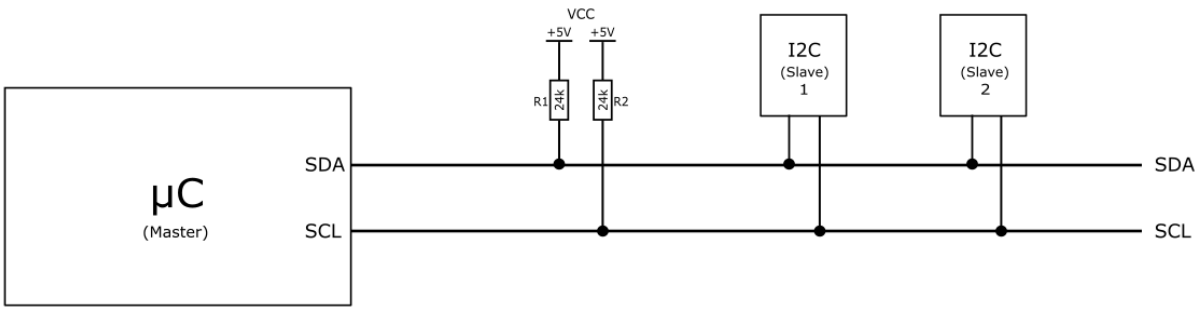
Dále se v této knihovně nachází příkaz na změření vzdálenosti, který vrací hodnotu v centimetrech, který je například použit v části programu, který detekuje objekt, a pokud je v menší vzdálenosti než 20 centimetrů, spouští další příkazy.

```
223  void loop()
224  {
225
226    while(UZSenzor.measureDistanceCm() < 20) {
```

15: Smyčka s podmínkou detekce objektu ultrazvukovým senzorem

2.7 OLED displej SSH1306

Displej OLED SSH1306 je dalším modulem na tomto robotovi, jedná se o modul s malou obrazovkou s rozlišením 128 na 64 pixelů. Tento modul stejně jako modul PCA9685 využívá ke komunikaci rozhraní I^{2}C. Pracuje na adrese 0x3C. A vzhledem k tomu, že na robotovi se již jedno zařízení s I^{2}C komunikací nachází, je proto potřeba udělat zapojení těchto zařízení za pomoci tzv. pull-up rezistorů neboli rezistorů zajišťujících logickou jedničku v obvodu. Tento



16: Diagram zapojení pull-up rezistorů v rozhraní I^{2}C

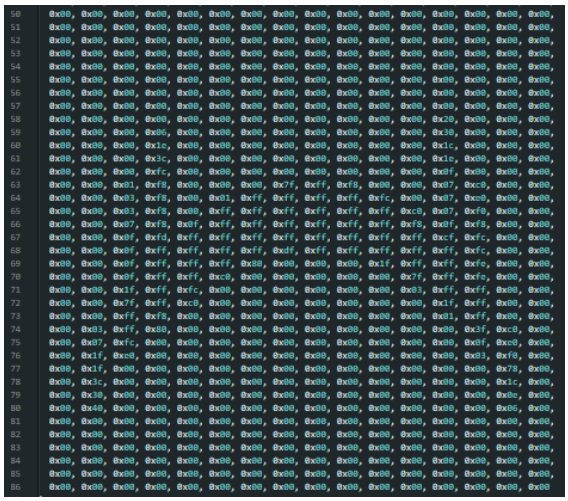
rezistor je umístěn mezi nosičem informace a napájením. V našem případě mají rezistory hodnotu něco kolem 24 kΩ, aby v obvodu nedošlo ke zkratu.

K naprogramování displeje je pro usnadnění zvolena knihovna Adafruit_SSD1306.h. V této knihovně je mnoho užitečných příkazů (begin(), display(), clearDisplay(), a mnoho dalších…). U tohoto robota používáme především výše zmíněné tři příkazy.

Příkazem begin() alokujeme paměť RAM, jakožto vyrovnávací paměť pro data (obrázky) a připíšeme piny a nastavení danému displeji. Dále příkazem display() zašleme data (obrázek) z paměti RAM na displej. Následně příkazem clearDisplay() displej vymažeme a uvedeme do původního stavu.

Pro zobrazení obrázku na displeji je potřeba obrázek převést do jazyka C++, na to existují webové aplikace, nebo mnoho dalších programů. Převedení obrázku vypadá takto:

17: Nákres úst robota vytvořený v malování



18: Úryvek nákresu úst převedený do C++

Modul s displejem použitý na tomto robotovi má pouze dvě barvy, to modrou žlutou, bohužel tyto barvy se nedají měnit, na displeji jsou dané části, kde se nachází jedna nebo druhá barva. Ovšem jednotlivých pixelů se dá regulovat jas.

2.7.1 Příprava obrázků

Příprava obrázků spočívá ve vytvoření obrázku v téměř jakémkoliv prostředí. Nejjednodušší volbou je například Malování – program, který se nachází na většině zařízení s operačním systémem Windows. V malování si nastavíme rozlišení plátna na velikost shodnou s velikostí displeje robota (128 x 64 pixelů), dále v jakémkoliv webovém rozhraní, které poskytuje funkci převádění ze souborů s příponou „.png“ na řadu hodnot v jazyce C++ (například https://javl.github.io/image2cpp/).

Výsledný kód poté vložíme do našeho programu a následně využijeme v knihovně Adafruit_SSD1306.h, která obrázek vyvolá na displeji.

2.8 Ovládání robota

Jak je již zmíněno v teoretické části této práce, ovládání robotů může být provedeno mnoha způsoby, v případě tohoto robota je ovládání provedeno skrze jednu z funkcí ESP-32, a to funkcí Bluetooth.

Aby programování jednotlivých pohybů nebylo tak náročné, byla vytvořena knihovna Mop.h, která velmi usnadňuje další programování robota. V knihovně se nachází v této době pár příkazů, ovšem v budoucnu se počet velmi rozroste s přibývajícími funkcemi robota. Zatím jsou zde příkazy jako StartPos() – uvede robota do základní polohy a Crouch() – skrčí robota do dřepu. Díky této knihovně v programu robota již nemusí být jednotlivě rozepsány tyto funkce a stačí knihovnu do programu naimportovat.

K jednoduššímu využití této funkce je zapotřebí knihovna BluetoothSerial.h, která je volně přístupná ke stažení na internetu. Za pomoci této knihovny ESP-32 dokáže komunikovat prostřednictvím sériového terminálu. A odesílat a přijímat zprávy ve formě textu.

Díky této funkci robot nemusí být připojen k ovladači pomocí drátů a může být ovládán na větší vzdálenost. Hlavním kladem této funkce je i to, že robot může být ovládán prostřednictvím jakéhokoliv zařízení s přístupem k Bluetooth – robota můžeme klidně ovládat pomocí chytrého telefonu.

Jedinou potřebnou věcí na zařízení, kterým chceme robota ovládat je funkce Bluetooth, přesněji možnost odesílat a přijímat určitý text přes sériový terminál. Pro získaní této možnosti

se dají využít aplikace pro systém Android, které tuto funkci poskytují. Existuje pár aplikací na tento způsob, které jsou volně ke stažení na platformě Obchod Play.

Také se dá tato funkce získat tím, že si aplikaci vytvoříme. Nezáleží tak moc jakým způsobem, či na jaké bázi je aplikace vytvořena, jediným požadavkem je, aby dokázala zpracovávat a odesílat data skrze síť Bluetooth, přesněji přes terminál.

2.8.1 Nastavení Bluetooth v programu robota

V programu robota se nastaví Bluetooth pomocí knihovny BluetoothSerial.h, která celý proces nastavení velmi usnadní. Nejprve je potřeba knihovnu nainstalovat a po instalaci je nutné knihovnu importovat. Dále na začátku programu je důležité definovat základní funkci knihovny. Pro komunikaci s aplikací si musíme zvolit hodnoty (data), které budeme zpracovávat, v našem případě to jsou hodnoty: 0, 1 a 2. Tyto hodnoty ovšem nejsou ukládány jako Integer (Int), ale ve formě char – znak. Je to převážně z důvodu, že pokud bychom chtěli komunikovat na jednoznakové bázi a použili bychom Integer, daly by se využít pouze hodnoty 0 – 9, ovšem v případě znaků existuje něco kolem 150 000 znaků, které se dají využít.

V programu aplikace se tedy nachází část, kterou nazývám „rozcestník dat“. Jedná se o část programu, ve které se přijmou data poslaná prostřednictvím Bluetooth a podle toho o jaká data se jedná, program vykoná činnost přiřazenou k jednotlivým znakům.

```
311  char mode = ' ';
312
313
314  if (Serial.available()) {
315    SerialBT.write(Serial.read());
316  }
317  if (SerialBT.available()) {
318
319    mode = SerialBT.read();
320
321    if (mode == '0')
322    {
323      waveMode = false;
324      scaredMode = false;
325      mop.StartPos();
326    }
327    if (mode == '2')
328    {
329      waveMode = true;
330      delay(2000);
331      scaredMode = false;
332      Serial.write("WaveMode True");
333    }
334    if (mode == '1')
335    {
336      scaredMode = true;
337      delay(2000);
338      waveMode = false;
339      Serial.write("ScaredMode True");
340    }
341
342  }
```

19: "Rozcestník dat“

2.8.2 Aplikace M. O. P.

Aplikace M. O. P. je mnou vytvořené prostředí pro ovládání robota. Aplikace je vytvořena za pomoci softwaru MIT App Inventor (http://appinventor.mit.edu/). Jedná se o bezplatný program na jednoduchou vizuální tvorbu aplikací a následné programování funkcí aplikace.

M. O. P. je odvozeno z názvu robota – Mechanicky Ovládaný Pavouk

Programování v tomto prostředí je velmi jednoduché – za pomoci blokového editoru. Na naučení se základů práce s programem stačí návody na YouTube, kterých je velmi mnoho dobře srozumitelných.

Aplikace M. O. P. se zatím skládá pouze ze čtyř funkčních tlačítek:

- **Domů** – uvede robota do základní polohy
- **Strach** – přepne robota do režimu strachu
- **Mávání** – robot vesele zamává
- **Bluetooth –** otevře seznam s výběrem spárovaných zařízení

Tyto tlačítka plní funkci odesílatele, a po stisknutí odešlou prostřednictvím sítě Bluetooth krátký, jednočíselný kód, pomocí kterého program nahraný do robota rozpozná, co za činnost má vykonat.

Pro připojení k robotovi je potřeba spárovat se s robotem, dále zmáčknout tlačítko Bluetooth, poté vyskočí seznam všech zařízení, se kterými je náš telefon (případně jiné zařízení) spárován a je potřeba vybrat to s názvem M. O. P. – tedy tohoto robota. Po chvilce by se text pod tlačítkem Bluetooth měl změnit na „připojeno“, tím je tento krok splněn.

Pro vykonání jednotlivých činností pak jen stačí zmáčknout daná tlačítka a robot se podle přijatých dat zachová.

V budoucnu je v plánu vizuální úprava aplikace a rozšíření možností činností robota. Dále by se také dala vytvořit univerzální webová aplikace, pro přístup k ovládání ze všech zařízení s přístupem k internetu a vhodným prohlížečem.

3 ZÁVĚR

V úvodu jsme si jako cíl práce nastavili sestrojit čtyřnohého robota mikrokontrolérem ESP32. Pro tuto úlohu jsme zvolili 3D tisk na vytvoření pevných částí robota, výsledkem práce je kvadruped poháněný dvanácti servomotory SG90, osazený ultrasonickým senzorem displejem.

Pro modelaci částí byl zvolen program Blender, což je program zcela zdarma vhodný pro začátečníky. Pro naučení se ovládání tohoto programu bylo využito sociální sítě Youtube, kde za pomoci návodů videí vysvětlením funkcí programu poté vznikly první 3D návrhy. Dále jako slicer pro 3D tiskárnu byl zvolen freeware Cura, který připraví 3D modely na tisk. Jako prostředí pro naprogramování robota byl zvolen program Arduino IDE, což je freeware velmi vhodný pro začátečníky. Pro nákresy obrázky užité této práci byly zvoleny programy Inkscape Microsoft Whiteboard, opět oba programy zdarma uživatelsky velmi přívětivé. Při vytváření robota bylo čerpáno převážně z předešlých zkušeností, dále ze zdrojů uvedených níže bibliografii.

Na ovládání servomotorů bylo nejdříve zvoleno přímé spojení mikrokontrolérem ESP32 přes GPIO piny za pomoci komunikace PWM. Jenže to se později ukázalo jako chaotické a nepřehledné, proto později plánování práce byl zvolen modul PCA9685, který převážně díky cable-managementu usnadnil orientaci ve vodičích zapojení robota.

Nejdříve byl robot osazen pouze servomotory, a zvládal vykonávat jen pohyb zadaný mikrokontrolérem, následně po osazení robota ultrasonickým senzorem jsme docílili získání odezvy robota na okolní události. Díky přidání displeje na robota se robot stal více přívětivý pro uživatele (převážně mladší, díky úsměvu).

V práci jsme si také vysvětlili průběh a chod programování a některé knihovny využité pro programování tohoto robota. Tento robot může sloužit jako model pro pozdější využití a zdokonalení.

Dále v budoucích pracích plánuji další rozvoj tohoto robota, přidání dálkového ovládání, přidání akumulátorů a zdokonalení mobilní aplikace pro jednoduché ovládání robota prostřednictvím funkce Bluetooth Wi-Fi. A případným využitím robota pro výuku robotiky na školách.

4 POUŽITÁ LITERATURA

1. **STUDNIČKA, Jiří.** *Vlastnosti mikrokontroléru ESP32.* Louny : Jiří Studnička.

2. **Systems, Espressif.** Espressif. [Online] 2022. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.

3. **Kotyrba, Eva Volná Martin.** Umělá inteligence. [Online] 2013. https://projekty.osu.cz/svp/opory/PrF_Volna,Kotyrba_Umela-intelig.pdf. ISBN 978-80-7464-330-9.

4. **FactoryAutomation.cz.** Průmyslové roboty: jaké jsou jejich druhy? *Magazín o průmyslové automatizaci a robotice.* [Online] FANUC Czech s.r.o., 2022. https://factoryautomation.cz/prumyslove-roboty-2/.

5. **Teja, Ravi.** How to Create ESP32 Web Server. *Electronics Hub.* [Online] https://www.electronicshub.org/esp32-web-server/.

6. **Trade Media International, s. r. o.** Vše průmyslu. [Online] Trade Media International, 2022. https://www.vseoprumyslu.cz/inspirace/produkty/rizeni-robotu-intuitivni-ovladani-robotu-diky-digitalnimu-dvojceti.html.

7. **portál, IoT.** IoT portál. [Online] https://www.iot-portal.cz/co-je-iot/.

8. **Wikipedia.** Wikipedia - API. [Online] https://cs.wikipedia.org/wiki/API.

9. **MakerBot Industries, LLC.** UltiMaker Thingverse. [Online] https://www.thingiverse.com/thing:1009659.

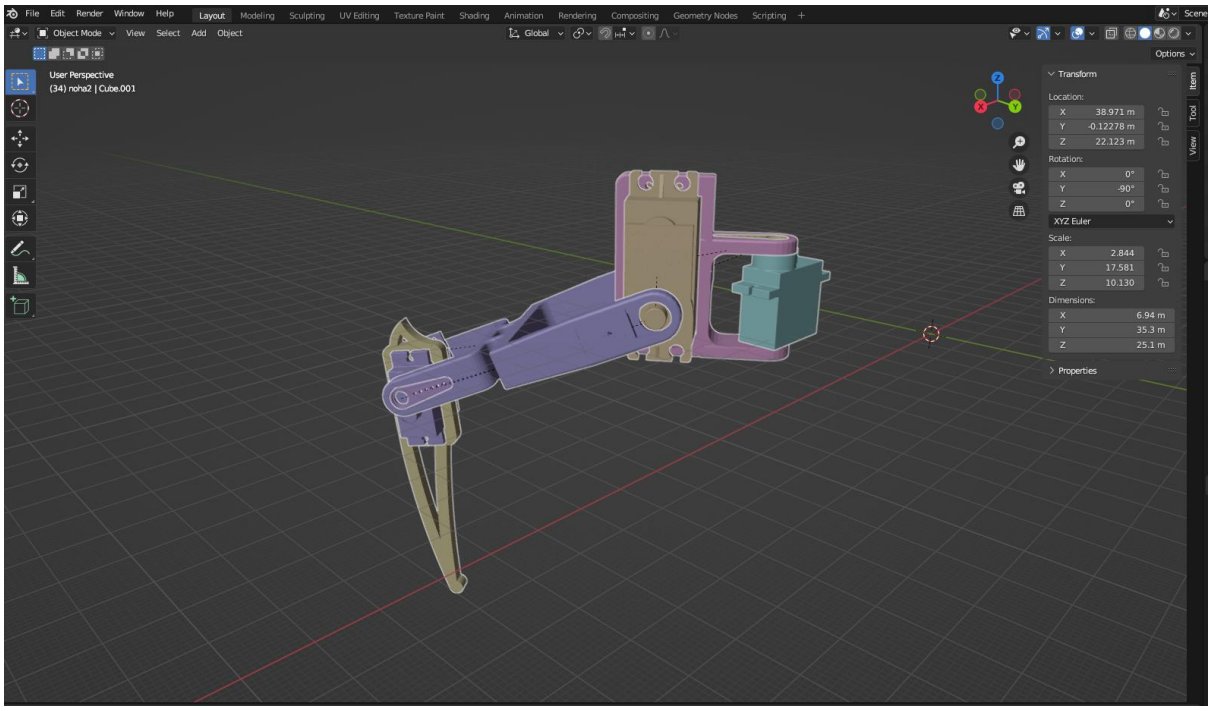
10. **Martinsos.** Arduino library for HC-SR04 ultrasonic distance sensor. *GitHub.* [Online] https://github.com/Martinsos/arduino-lib-hc-sr04.

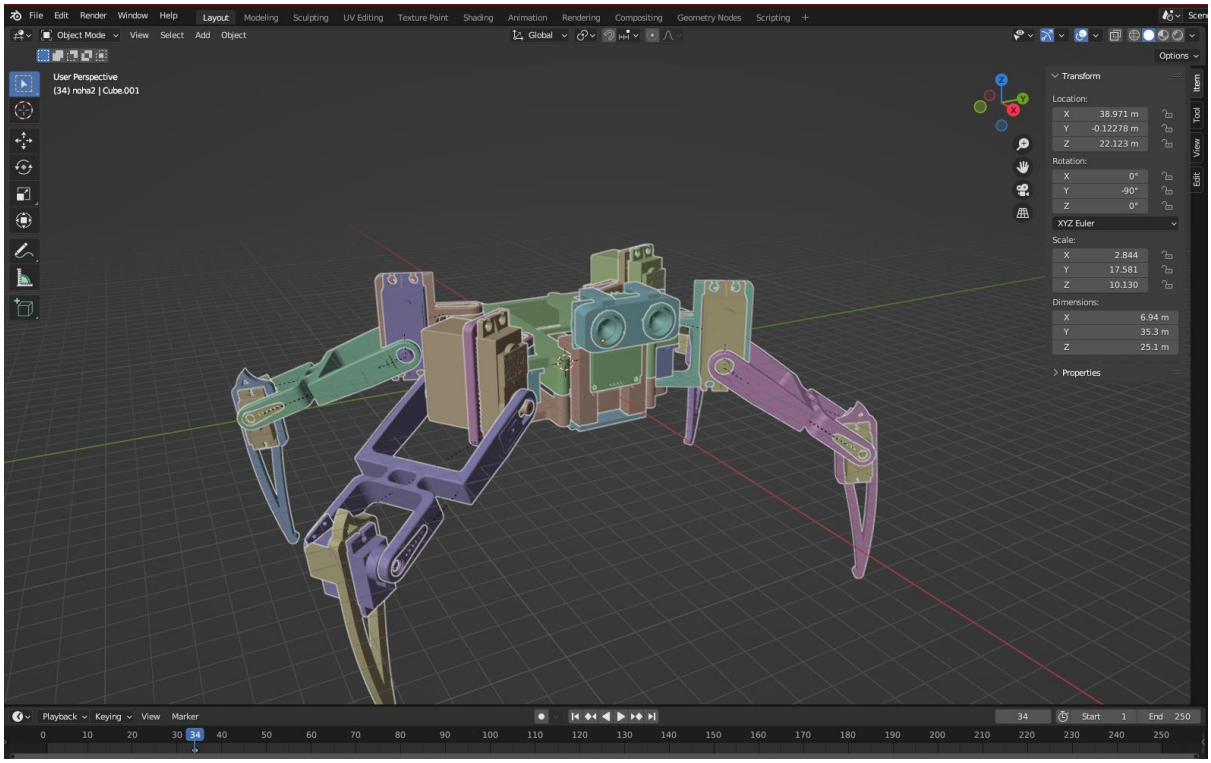
11. **Keyestudio.** ***Keyestudio.*** **[Online] 20. 6 2019. https://wiki.keyestudio.com/File:0413%E5%9B%BE%E7%89%871.png.**

5 SEZNAM OBRÁZKŮ

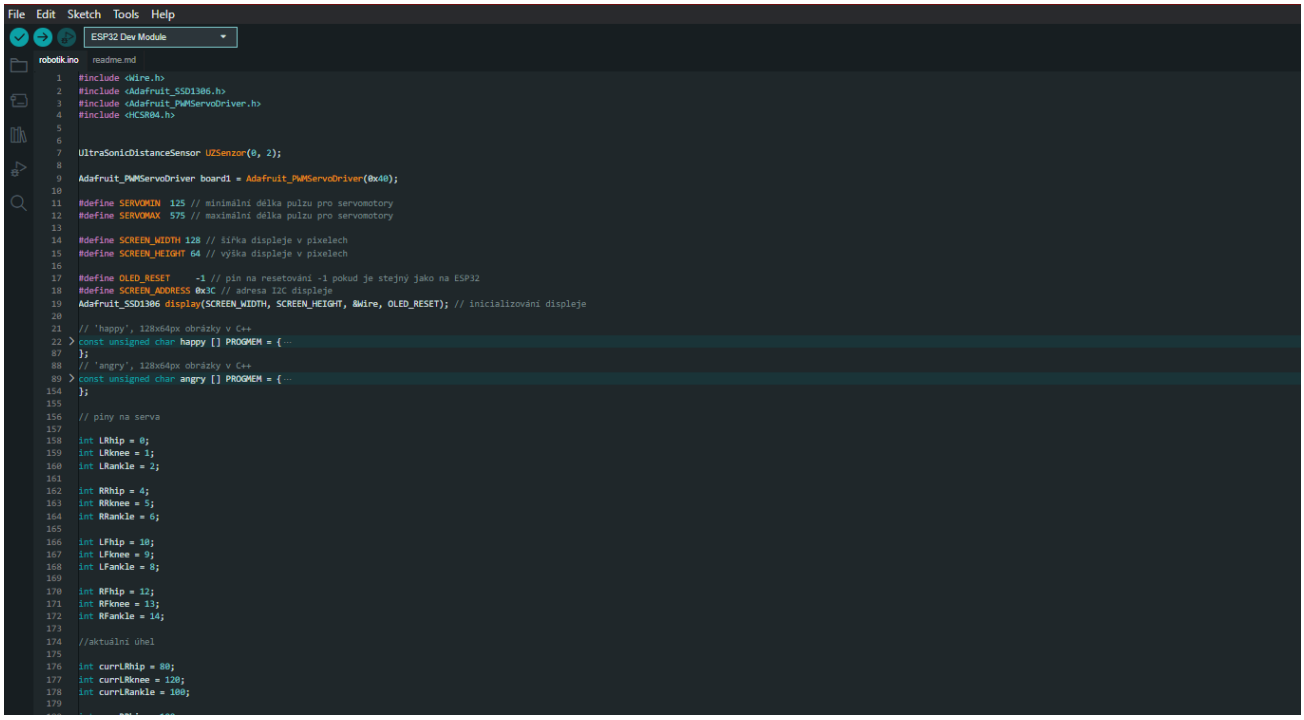
Všechny použité obrázky pochází z archivu autora.

6 PŘÍLOHA 1: BLENDER - MODELOVÁNÍ ČÁSTÍ ROBOTA

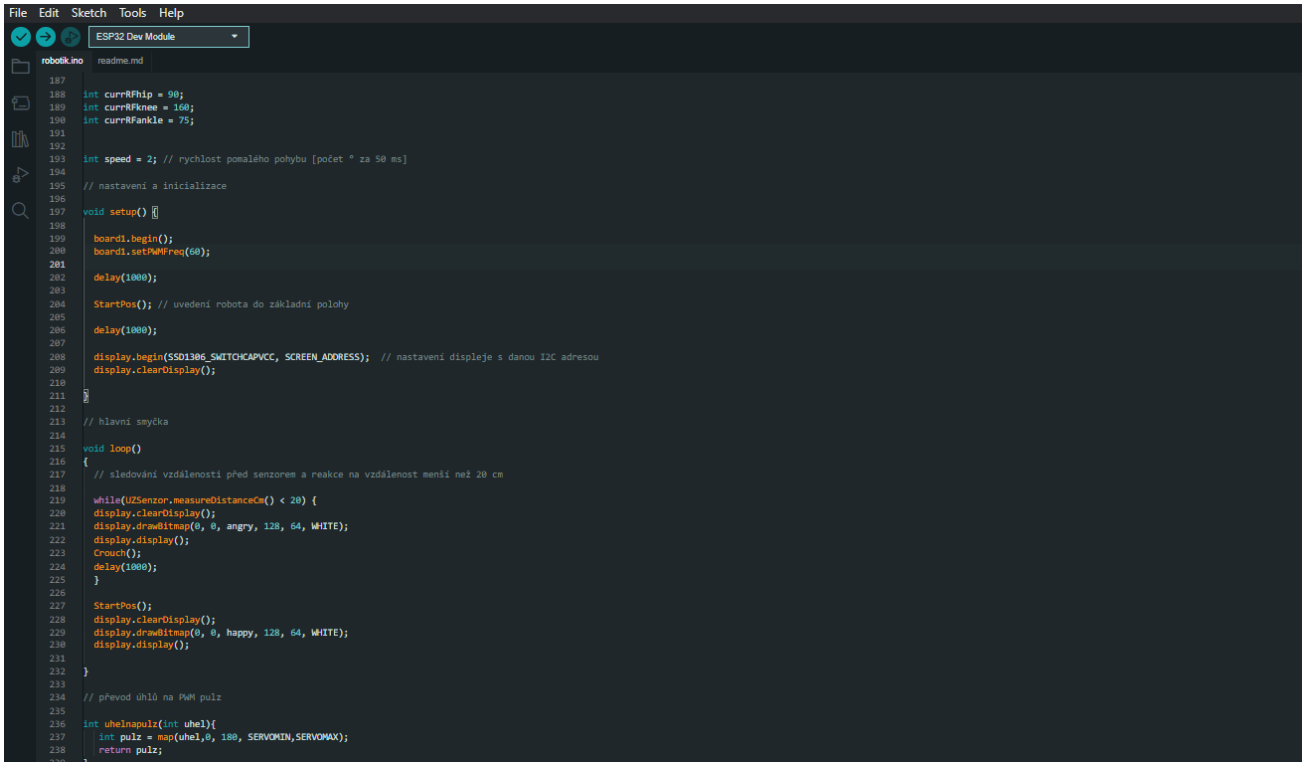




PŘÍLOHA 2: ARDUINO IDE – PROGRAMOVACÍ PROSTŘEDÍ

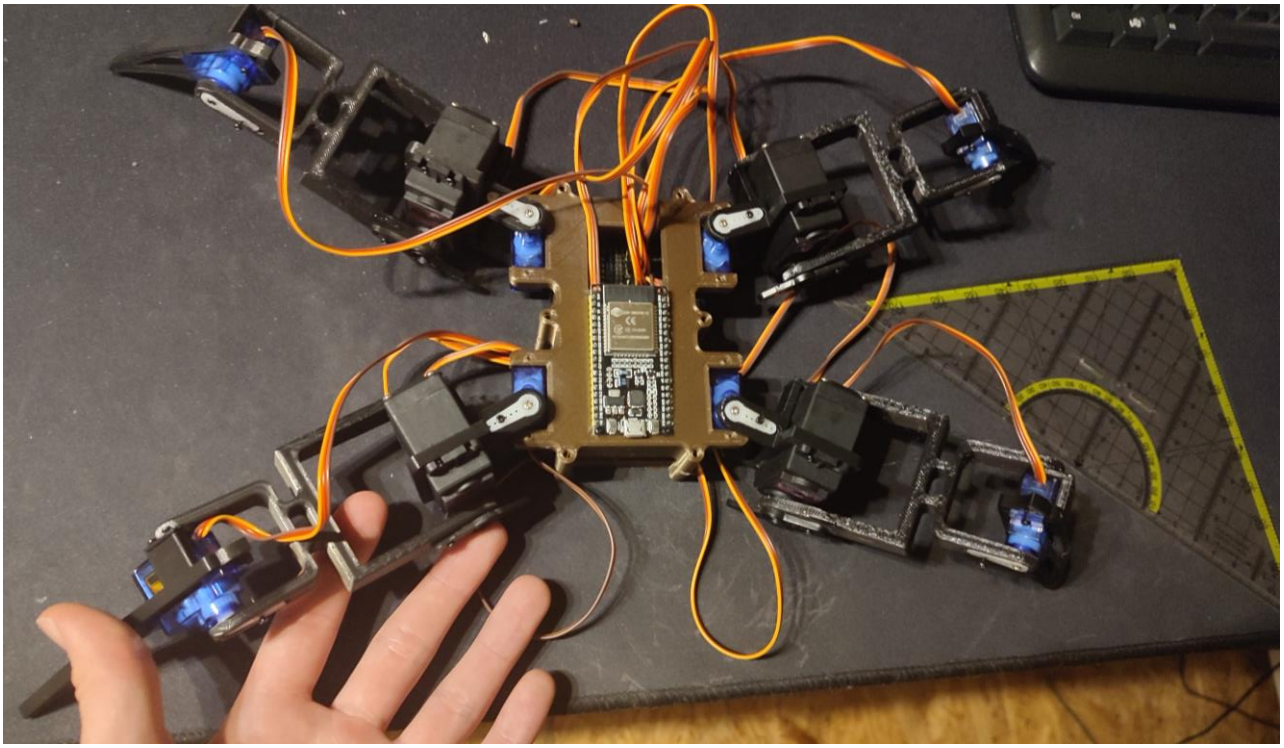


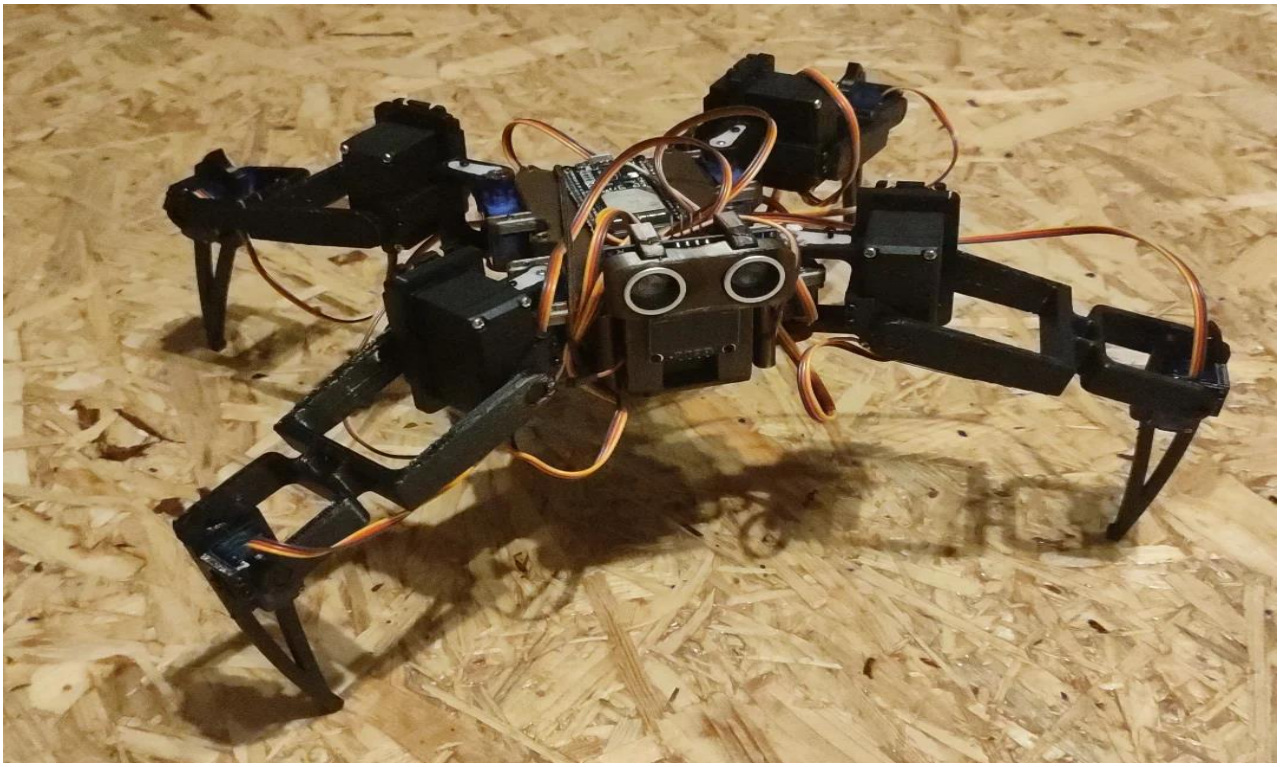
```
File  Edit  Sketch  Tools  Help
ESP32 Dev Module
robotik.ino  readme.md
#include <Wire.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_PWMServoDriver.h>
#include <HCSR04.h>


UltraSonicDistanceSensor UZSenzor(0, 2);

Adafruit_PWMServoDriver board1 = Adafruit_PWMServoDriver(0x40);

#define SERVOMIN  125 // minimální délka pulzu pro servomotory
#define SERVOMAX  575 // maximální délka pulzu pro servomotory

#define SCREEN_WIDTH 128 // šířka displeje v pixelech
#define SCREEN_HEIGHT 64 // výška displeje v pixelech

#define OLED_RESET     -1 // pin na resetování -1 pokud je stejný jako na ESP32
#define SCREEN_ADDRESS 0x3C // adresa I2C displeje
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET); // inicializování displeje

// 'happy', 128x64px obrázky v C++
const unsigned char happy [] PROGMEM = {…
};
// 'angry', 128x64px obrázky v C++
const unsigned char angry [] PROGMEM = {…
};

// piny na serva

int LRhip = 0;
int LRknee = 1;
int LRankle = 2;

int RRhip = 4;
int RRknee = 5;
int RRankle = 6;

int LFhip = 10;
int LFknee = 9;
int LFankle = 8;

int RFhip = 12;
int RFknee = 13;
int RFankle = 14;

//aktualni úhel

int currLRhip = 80;
int currLRknee = 120;
int currLRankle = 100;
```



```
File  Edit  Sketch  Tools  Help
ESP32 Dev Module
robotik.ino  readme.md

int currRFhip = 90;
int currRFknee = 160;
int currRFankle = 75;


int speed = 2; // rychlost pomalého pohybu [počet ° za 50 ms]

// nastaveni a inicializace

void setup() {

  board1.begin();
  board1.setPWMFreq(60);

  delay(1000);

  StartPos(); // uvedeni robota do základní polohy

  delay(1000);

  display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS);  // nastavení displeje s danou I2C adresou
  display.clearDisplay();

}

// hlavní smyčka

void loop()
{
  // sledování vzdálenosti před senzorem a reakce na vzdálenost menší než 20 cm

  while(UZSenzor.measureDistanceCm() < 20) {
  display.clearDisplay();
  display.drawBitmap(0, 0, angry, 128, 64, WHITE);
  display.display();
  Crouch();
  delay(1000);
  }

  StartPos();
  display.clearDisplay();
  display.drawBitmap(0, 0, happy, 128, 64, WHITE);
  display.display();

}

// převod úhlů na PWM pulz

int uhelnapulz(int uhel){
  int pulz = map(uhel,0, 180, SERVOMIN,SERVOMAX);
  return pulz;
```

PŘÍLOHA 3: SESTAVENÝ ROBOT





PŘÍLOHA 4 – SCHÉMA ZAPOJENÍ ROBOTA